IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 30, 2020, accepted February 4, 2021, date of publication February 16, 2021, date of current version March 3, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3059806

Image-Based Scam Detection Method Using an Attention Capsule Network

LINGYU BIAN[1], LINLIN ZHANG[1], KAI ZHAO[1], HAO WANG[2], AND SHENGJIA GONG[1]
[1]School of Cyber Science and Engineering, College of Information Science and Engineering, Xinjiang University, Urumqi 830046, China
[2]School of Software, Xinjiang University, Urumqi 830046, China

Corresponding author: Linlin Zhang (zllnadasha@xju.edu.cn)

This work was supported in part by the Natural Science Foundation of Xinjiang Uygur Autonomous Region under Grant 2019D01C062, Grant 2019D01C041, and Grant 2020D01C028; in part by the Graduate Research Innovation Project of Xinjiang Uygur Autonomous Region under Grant XJ2020G065 and Grant XJ2019G063; in part by the National Innovation Training Project for College Students under Grant 201910755030; in part by the National Natural Science Foundation of China under Grant 12061071; and in part by the Higher Education of Xinjiang Uygur Autonomous Region under Grant XJEDU2017M005 and Grant XJEDU2020Y003.

ABSTRACT In recent years, the rapid development of blockchain technology has attracted much attention from people around the world. Scammers take advantage of the pseudo-anonymity of blockchain to implement financial fraud. The Ponzi scheme, one of the main scam methods, has defrauded investors of large amounts of money, thereby harming their interests and hindering the application of blockchain. Unfortunately, the current detection technology typically largely relies on the source code of the contract or uses a single feature which does not fully represent the contract characteristics. In such a case, the detection of Ponzi schemes with high efficiency becomes urgent. In this paper, we propose an image-based scam detection method using an attention capsule network (SE-CapsNet) focused on Ethereum. The sequence of bytecode, the opcode frequency, and the application binary interface (ABI) call are extracted as features from the contract bytecode and ABI, further converted into grayscale images, and then mapped into three color channels to generate RGB images, which are used as the input of the model for detecting the Ponzi scheme contract. In addition, we employ fancy PCA for data augmentation to reduce the impact of imbalanced data on the detection results. Experimental results show that the image-based detection method using deep learning models can effectively detect contracts before transactions occur. Among them, our proposed SE-CapsNet obtains great detection results, with an F1 score of 98.38%.

INDEX TERMS Blockchain, capsule network, Ethereum, Ponzi scheme, smart contract.

I. INTRODUCTION

After decades of development, blockchain has emerged as a technology with a wide range of applications, and it has attracted extensive attention from both academia and industry, especially in the field of cryptocurrency, where market valuations such as Bitcoin and Ether are rising at increasing rates. Under the lure of huge profits, due to the pseudo-anonymity of blockchain technology, scammers hidden behind pseudonymous accounts can easily complete cryptocurrency transactions as normal traders without their true intentions being identified [1]. Once a financial scam occurs, it is difficult to track, let alone take countermeasures or even recover property, and this hurts invertors heavily.

The associate editor coordinating the review of this manuscript and approving it for publication was Dongxiao Yu.

Currently, the situation is worsening with increasing fraud taking place in blockchains. According to the latest research report published by Chainalysis [2], a blockchain analysis company, the total value of defrauded cryptocurrency was as high as 4.3 billion US dollars in 2019, and most of it came from Ponzi schemes (up to 92%). The Ponzi scheme is a typical well-known type of pyramid scheme that usually promises high rates of return with little risk for investors to create the illusion of making money [3]. However, most investors are unable to identify scams, and once they invest, the economic losses caused are generally irreversible. Therefore, to some extent, we can say that the Ponzi scheme has damaged the reputation of the whole blockchain ecosystem, including Ethereum.

Ethereum is an open-source and blockchain-based decentralized platform that enables programmers, as well as

scammers, to create versatile smart contracts and decentralized applications [4]. That is, scammers can easily create a Ponzi scheme. In recent years, the number of Ponzi schemes has increased daily. Many famous Ponzi schemes, such as PlusToken, Forsage, and FairWin, can be found on Ethereum. Investors have lost hundreds of millions of dollars to these Ponzi schemes. Hence, it is an urgent task to detect Ponzi schemes on Ethereum.

Early Ponzi schemes could be found in the investment advertisements of the Ethereum community forum. Ethereum uses an account-based model, which contains two types of accounts. One is an externally owned account, and the other is the so-called contract account, which will be shortened as a contract in the following text. In this case, the behavior of a scammer is often embodied by the externally owned account and its related transactions. Therefore, corresponding features can be extracted based on historical transactions to detect Ponzi schemes. Nonetheless, this method cannot be performed in real time, and it requires a large amount of relevant transaction information, such as a full dataset.

Smart contracts are programs that execute autonomously without a third party on Ethereum [5]. Over time, the implementation of Ponzi schemes as smart contracts has gradually become more popular. Scammers use this auto-executive feature of smart contracts to publish a convincing project plan, which shows that investors can obtain bonuses in time, to gain investor trust. Therefore, we can detect Ponzi schemes by analyzing the source codes of the smart contracts before transactions occur, and this could prevent the widespread use of such Ponzi schemes. However, the following challenges regarding the analysis of Ponzi scheme contracts still exist.

1) Insufficient source code. According to Etherscan [6], only approximately 1% of smart contracts have available Solidity source code [7]. In cases with insufficient contract source code, the selection of appropriate features directly affects the performance of the detection method. In addition, how to express the selected features also needs to be carefully considered.
2) Low accuracy. The existing research works have proven that deep learning technology is a feasible method in the field of smart contract classification [8]. Since the code length of a smart contract is short, we need to choose an appropriate deep learning model because it may directly affect the accuracy of our experiments.

The purpose of this paper is to design a novel detection method that can detect not only Ponzi schemes but also additional types of scams in the future. Therefore, we propose an image-based scam detection method using an attention capsule network (SE-CapsNet) based on Ethereum. First, based on the contract address of the Ponzi scheme, the bytecode and application binary interface (ABI) of the corresponding contract are downloaded as basic features. Then, three types of features are extracted and converted into grayscale images. Next, they are merged into an RGB image as an input for the model to complete the Ponzi scheme contract detection process. The main contributions of this detection task are divided into the following four aspects.

1) Most contracts have bytecode and ABI. By analyzing the bytecodes and ABIs of both Ponzi scheme contracts and non-Ponzi contracts, the problem of the lack of source code in practical applications can be solved. Once the contract is deployed, we can check whether the contract is a Ponzi scheme or not, hence the losses incurred by investors can be reduced.
2) After years of research, malware detection combined with code visualization has proven to be an efficient and capable detection method. Based on the downloaded bytecode and ABI, the bytecode sequence, the opcode frequency sequence, and the ABI call sequence are obtained, and the above three features are combined to generate RGB images. In this way, we can improve the problem that a single feature cannot comprehensively represent the characteristics of a Ponzi scheme contract.
3) We use fancy PCA to enhance the data of Ponzi scheme images, and we obtain a total of 1,600 Ponzi scheme images to form a relatively balanced dataset. By using such a method, the impact of extremely imbalanced data on the detection results can be reduced.
4) We combine the Squeeze-and-Excitation (SE) block and capsule network to detect Ponzi schemes. The SE block has a simple structure, and the accuracy of the experiment can be improved by calculating the channel attention of the image. The capsule network can capture additional information, is suitable for a small dataset, and is proven to detect Ponzi scheme contracts efficiently on Ethereum.

The remainder of this paper is organized as follows. Section II introduces the research trends of related fields from three viewpoints. In Section III, we emphatically elaborate on the image-based scam detection method using the SE-CapsNet proposed in this paper; it is divided into three modules and introduced in detail. We evaluate the proposed method in Section IV and summarize the study in the last section of the paper.

II. RELATED WORK

A. SCAM ACCOUNT DETECTION

The ecosystem of Ethereum shows that the phenomenon of scamming is becoming increasingly serious, and the detection of scams is imminent. Existing research works [9], [10] have focused on extracting features from the transaction history information of externally owned accounts to detect scams. For example, Wu *et al.* [11] proposed a novel algorithm named trans2vec based on transaction amount and timestamp, was used to extract features, and it combined with a one-class support vector machine to detect phishing on Ethereum. Toyoda *et al.* [12] extracted the frequencies of patterns as key features. The experimental results established that approximately 83% of HYIP accounts (belonging to a Ponzi scheme) can be correctly classified using the XGBoost

algorithm. Bartoletti *et al.* [13] extracted a total of 32 features from the transaction history of Bitcoin, used downsampling to balance the dataset, and combined supervised machine learning algorithms to detect Ponzi schemes. Subsequently, Bartoletti *et al.* [14] identified a Ponzi scheme on Ethereum by analyzing the number of transactions and transaction amounts, and then they evaluated the impact of the Ponzi scheme.

There are certain deficiencies in the current scam detection methods for externally owned accounts, because a large amount of transaction data is required as the basis for feature extraction, and real-time detection is inferior. Therefore, related research based on contract codes has gradually emerged. Torres *et al.* [15] employed a symbolic execution approach, defined a heuristic method for automatically detecting honeypot contracts, and analyzed honeypot contracts from the perspectives of behavior, diversity, and activity. Chen *et al.* [16] collected smart contracts to obtain bytecode and built a control flow graph (CFG), which was used to identify non-deployable contracts and help explore contract transaction rules. Chen *et al.* [17] extracted features from the transaction histories and opcodes of smart contracts, and established a smart Ponzi scheme classification model. Jung *et al.* [18] proposed full-feature model, which combined the Gini time features based on transaction behaviors with the opcode features of contract to detect Ponzi schemes.

In summary, by analyzing the opcode and other features of a smart contract code, it is possible to analyze whether the logic of the code is similar to those of Ponzi scheme contracts, and this approach plays a positive role in improving the detection results. So, we use the bytecodes and ABIs owned by most contracts instead of source code to detect Ponzi schemes.

B. CODE VISUALIZATION

In more recent years, code visualization has been widely adopted in malware classification tasks, providing an end-to-end detection method that can effectively process data samples; this technique has obtained remarkable classification results [19]. In 2011, Nataraj *et al.* [20] proposed converting binary code to the values of pixels to generate a grayscale image for the classification of malware. Since then, code visualization has received widespread attention from researchers. Cui *et al.* [21] converted malicious code into grayscale images, and the problem of imbalanced data was solved using the bat algorithm. Finally, the features of the malware images were extracted automatically by a convolutional neural network, which classified the images with accuracy reaching 94.5%. Cui *et al.* [22] further improved upon their work by using the non-dominated sorting genetic algorithm II (NSGA-II) to obtain a better classification effect. Naeem *et al.* [23] designed a method to convert binary malware files into grayscale images and used two patterns, local and global, to classify malware. From the experimental portion of their study, the classification accuracy reached 98.4%.

The conventional methods of code visualization select a single feature, for example, binary data, which could not completely reflect the structural characteristics of code. Research related to image enhancement has slowly emerged over time. Sun *et al.* [24] enhanced malicious code images by filling binary data, ASCII character information, and PE structure information into three channels to form an RGB image. This method increased the interpretability of the model and improved the accuracy and robustness of code detection. Fang *et al.* [25] mapped the entropy, bytecode and proportion of sections in a DEX file to the red channel, green channel and blue channel of an RGB image. Then, the color and texture of the image and text were extracted as features. Eventually, the F1 score of the familial malware classification was 96%.

The approach based on image enhancement not only improve the accuracy of detection but also solve the problem of insufficient information resulting from the use of a single feature. Therefore, we extract various features based on bytecodes and ABIs of smart contract and choose the method of image enhancement to complete the detection task.

C. SE BLOCK AND CAPSULE NETWORK

This paper intends to combine the SE block with the capsule network. In the related field of computer vision, attention mechanisms are divided primarily into the spatial domain and channel domain. The spatial domain is mainly used to extract the key information of an image, while the channel domain focuses on the weight of the new channel generated by the image through the convolution kernel. The SE block [26] belongs to the attention mechanism of the channel domain, and it is mainly used to learn the relationships between channels with the goal of obtaining the channel attention weight. Moreover, the SE block is simple and compact with respect to the calculation, and it has also achieved outstanding performance on the ImageNet classification task. Yu [27] combined the SE block with the bottleneck layer for the classification of retinal diseases. The experimental results showed that the classification accuracy had been significantly improved with a few network parameters.

The capsule network (CapsNet) was proposed by the Hinton team in 2017 [28]. Although a convolutional neural network has translation invariance, the pooling layer still causes information loss. The capsule network is based on structures called capsules and uses dynamic routing algorithms between capsules. The relationships between low-level features and high-level features can be well represented by the capsule network, which can obtain a good classification effect with a small amount of data. The capsule network consists primarily of a convolutional layer, a PrimaryCaps layer and a DigitCaps layer.

Research has shown that the capsule network performs well in image classification tasks, especially when the differences among classes are small [29]–[31]. However, the capsule network is not suitable for the classification of complex images. Given this situation, researchers have improved the capsule network according to application requirements.

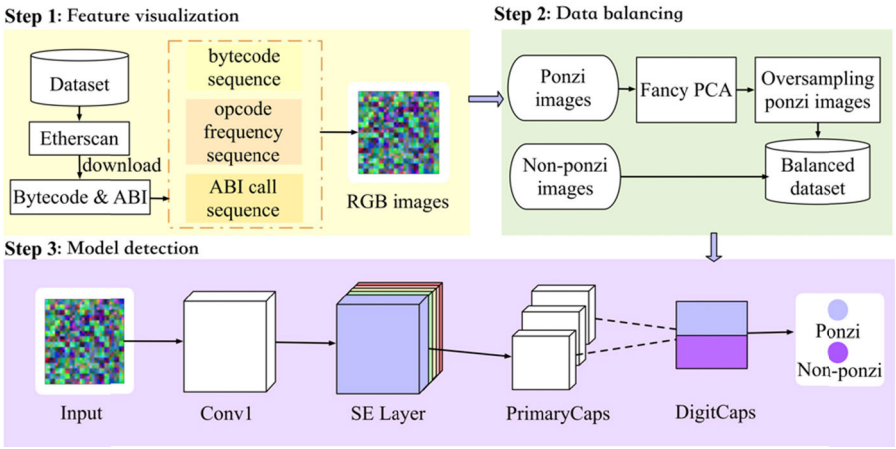

FIGURE 1. The framework of Ponzi scheme contract detection.

Xiang *et al.* [32] proposed MS-CapsNet, which added multi-scale capsule encoding units behind the convolutional layer. These units can extract different levels of features and encode them into primary capsules of different dimensions, and the improved dropout can enhance the robustness of the capsule network. Cheng *et al.* [33] proposed two structures, Cv-CapsNet and Cv-CapsNet++, to obtain complex-valued features and complex-valued capsules. At the same time, the dynamic routing algorithm was extended to the complex-valued domain. Compared to other existing methods, this method requires fewer trainable parameters and can be adapted to complicated datasets.

It can be seen that improving the CapsNet for different application requirements is a feasible research method. Considering that the number of samples in this paper is insufficient, the image structure is not too complicated; furthermore, the accuracy needs to be improved, meanwhile the detection speed should not be too slow, we employ the SE block to improve the structure of the capsule network to properly fit the detection objects in this paper.

III. PROPOSED METHOD

Fraud continues to increase year by year, especially with an increasing number of Ponzi scheme contracts appearing on Ethereum; thus, an image-based scam detection method using SE-CapsNet is proposed. The detection samples in this paper are Ponzi scheme contracts and non-Ponzi scheme contracts. In the following, we refer to the Ponzi scheme and non-Ponzi scheme as Ponzi and non-Ponzi. The method consists of three modules, which are feature visualization, data balancing, and model detection, respectively. The framework of the proposed Ponzi scheme contract detection method is shown in Figure 1. See the following Algorithm 1 for details.

A. FEATURE VISUALIZATION

Developers use the high-level programming language Solidity [34] to write the code of smart contracts and compile smart contracts that include code logic through a compiler, while at the same time obtaining bytecodes and ABIs. They then deploy smart contracts on the Ethereum network. So, we can easily get the bytecodes and ABIs of almost all contracts. The existing scam contract detection methods mainly use a single feature, which has limited ability to express the content of the contract.

Feature visualization refers to the extraction of three types of features based on bytecodes and ABIs, including bytecode sequences, opcode frequency sequences and ABI call sequences. The sequence of bytecodes usually used in static analysis methods can reflect the semantic information of the contract. The frequency sequence of opcodes can indicate the most frequent operations of the contract. The sequence of ABI calls can represent the context information of the calling contract. Using these features will greatly help the accuracy of detection. Then, the corresponding images are generated, and these images are finally combined into RGB images. The RGB images have three color channels, and these channels can store information and intuitively visualize contracts. In addition, because the images generated by similar contracts are also similar, so we can detect scam based on images.

1) BYTECODE SEQUENCE

A bytecode is composed of a string of hexadecimal digits, and this makes it difficult to read and understand the bytecode directly. Therefore, we use the code visualization method to convert the bytecodes into pixels, which can reduce the feature extraction time. We convert the bytecode into a decimal integer in order and employ the digit 255 to represent white and 0 to represent black; therefore, a grayscale image can be generated that shows the sequence of a given bytecode. Figure 2 demonstrates the process of producing bytecode sequence image.

2) OPCODE FREQUENCY SEQUENCE

Our work uses the Ethereum EVM bytecode disassembler named ethereum-dasm [35] to convert bytecodes into

Algorithm 1 Image-Based Scam Detection Method Using an Attention Capsule Network

Input: Ponzi scheme dataset: download the bytecode (B_i) and ABI (A_i) of each contract i;
Output: Detection results: accuracy, precision, recall, and F1 score;

Step 1: Feature visualization
  While (A_i != null) and (B_i != null) **do**
    bytecode sequence = bytearray (B_i);
    $Image\_BY_i$ = visualization (bytecode sequence); // This method means every eight-bit binary number is converted to a decimal number and then mapped into an image
    opcode sequence = disassembly (B_i);
    opcode weight = TF-IDF (opcode sequence);
    opcode frequency sequence = Simhash (opcode weight);
    $Image\_OP_i$ = visualization (opcode frequency sequence);
    ABI call sequence = DFS (A_i);
    get binary file after PV-DM (ABI call sequence);
    $Image\_AB_i$ = visualization (binary of ABI call sequence);
    $Feature\_image_i$ = merge ($Image\_BY_i$, $Image\_OP_i$, $Image\_AB_i$);
  End While
End Step 1
Step 2: Data balancing
  New images = Fancy PCA($Feature\_image_K$); // K stands for Ponzi contracts
  New dataset = $Feature\_image_i$+ New images;
End Step 2
Step 3: Model detection
  Add SE block after the convolutional layer of CapsNet to build the SE-CapsNet;
  Train the SE-CapsNet;
  Input the test samples into the trained network to detect;
End Step 3
Return the detection result.

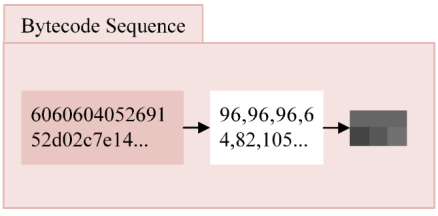

FIGURE 2. The process of producing bytecode sequence.

opcodes. EVM uses single-byte opcodes, which means that each byte represents an operation or instruction. Based on the term frequency–inverse document frequency (TF-IDF) algorithm, we want to obtain opcodes that are frequently used and helpful for the Ponzi contract detection task. To obtain a square image, we select the 32 opcodes with the largest TF-IDF weights from the disassembly results. The calculation formulas of TF-IDF are as shown in (1) - (3):

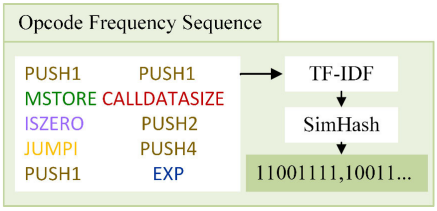

FIGURE 3. The process of generating opcode frequency sequence.

$$TF - IDF_{i,j} = TF_{i,j} * IDF_i \tag{1}$$

$$TF_{i,j} = \frac{N_{i,j}}{\sum_k N_{k,j}} \tag{2}$$

$$IDF_i = \log\left(\frac{D}{D_i + 1}\right) \tag{3}$$

In this paper, we disassemble each contract to obtain the corresponding opcode, thus forming an opcode document for each contract. Here, TF stands for term frequency, the numerator represents the number of occurrences of opcode i in document j, and the denominator is the sum of occurrences of all opcode k in document j. IDF stands for inverse document frequency, D stands for the total number of all opcode documents, and D_i stands for the total number of documents that contain opcode i.

SimHash is a locality sensitive hash algorithm [36], which means that similar inputs yield similar outputs to maintain data similarity. SimHash is used to extract the binary information of the 32 opcodes by calculating the hash value of each opcode through the hash function. Using this method, each opcode can be encoded into binary code with a fixed length of 64 bits. Then, all the binary codes representing the opcodes are linked together in groups of eight binary numbers for conversion to decimal numbers to generate the image of the opcode frequency sequence. The image generation process is the same as the steps for generating bytecode sequence images. Figure 3 shows the process of generating an image for the opcode frequency sequence.

3) ABI CALL SEQUENCE

The ABI is the standard way to interact with contracts in the Ethereum ecosystem, and it is similar to the API of the application. The calling relationship between functions and events can be expressed effectively by the ABI, as we usually believe that the ABI call sequences of the scam contract are unlike with those of regular contracts. By extracting the ABI call sequence from a smart contract, we can discover the contextual relationships between ABIs. Therefore, we choose the ABI call sequence as a feature in this paper.

First, we traverse the ABI of each contract with depth-first searches and obtain the calling order between ABIs. Then, the

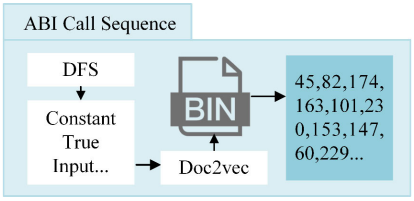

FIGURE 4. The process of generating ABI call sequence.

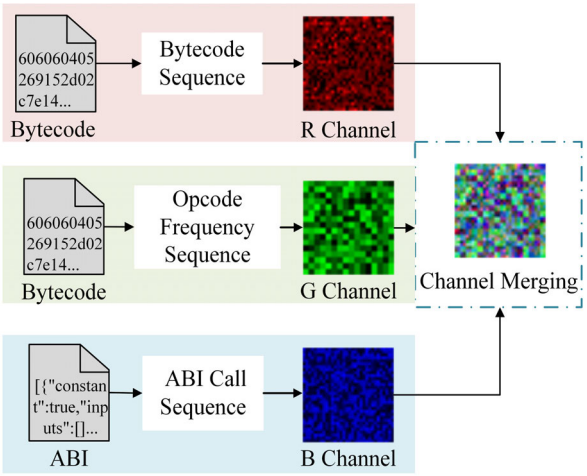

FIGURE 5. RGB image generation.

corresponding word segmentation task is performed. Next, we select the distributed memory model of paragraph vectors (PV-DM) [37] to generate paragraph vectors that can be used as the feature. In particular, the model adds a paragraph token that maps each paragraph to a unique vector. Each ABI sequence after text processing is regarded as a paragraph, and the paragraph vector of each ABI is obtained after training. Considering that the generated image cannot be too small, we aim to generate a 1024-dimensional paragraph vector for each ABI sequence, store it as a. *bin* file, and then convert the binary number to a decimal. After image visualization is completed, the ABI call sequence composed process is shown in Figure 4.

4) IMAGE MERGING

The three types of sequences reflect features of three different states. By mapping the bytecode sequence on the R channel, the opcode frequency sequence on the G channel, the ABI call sequence on the B channel, and finally merging the three channels together to form an RGB image, our method not only combines multiple features reasonably and orderly but also compensates for the shortcomings of single features to some extent. Figure 5 illustrates how an RGB image is merged.

B. DATA BALANCING

The data of scam contracts are usually extremely unbalanced. In most cases, contracts are normal. Therefore, the number of Ponzi contracts is very small compared to the number of normal contracts. Thus, there is a data imbalance problem between the Ponzi images and the non-Ponzi images generated in this paper. Unlike in the field of image recognition, the images produced in this paper must have their integrity ensured. Fancy PCA (PCA jittering) has the advantages of simplicity and efficient operation [38], so it is suitable for the rapid augmentation of Ponzi contracts.

Fancy PCA performs principal component analysis on the pixel values of RGB images to obtain a $3 * 3$ covariance matrix, calculates the corresponding eigenvalues and eigenvectors, and then arranges them in descending order. The augmented image is composed of the pixel value of the original RGB image plus the result of the following formula. The calculation formula is shown as (4):

$$[P_1, P_2, P_3]\,[\alpha_1\lambda_1, \alpha_2\lambda_2, \alpha_3\lambda_3]^{\mathrm{T}} \tag{4}$$

where P_1, P_2, P_3 are the eigenvectors, $\lambda_1, \lambda_2, \lambda_3$ represent the eigenvalues, and α_i is a random variable drawn from a Gaussian with mean 0 and standard deviation 0.1 in the original paper. The new Ponzi images are generated by adjusting the standard deviation parameters together with the original sample of Ponzi images to form a new Ponzi dataset, further achieve the effect of data balancing, and finally improve the detection effect of the proposed method.

C. MODEL DETECTION

The applications of deep learning in the related field of malware detection have achieved excellent research achievements; in particular, the capsule network considers the relationships between features, and this approach has advantages when applied to small samples. This paper combines a channel attention mechanism, called the SE block, with the capsule network to form the SE-CapsNet model, which is mainly composed of the following four layers.

1) CONVOLUTIONAL LAYER

The first layer is a simple convolutional layer designed to extract local features using 3×3 convolution kernels with a step size of 1 in combination with the ReLU activation function.

2) SE LAYER

The SE block is simple to use, it can improve the feature extraction ability of the model, and it is conducive to classification, which consists of two operations: squeeze and excitation. The purpose of squeeze operation is to obtain the global features of a given channel. u_C represents the C-*th* feature map, which is output by the convolutional layer. Through global average pooling, we can obtain channel-wise statistics z_C. Excitation operation is the process of learning channel weights, where σ represents the sigmoid activation function, δ denotes the ReLU function, and W_1 and W_2 are the dimensionality-reducing and dimensionality-increasing actions, respectively. Through the excitation operation, we can quickly learn a nonlinear interaction between channels and finally obtain the learned channel weights s. Finally,

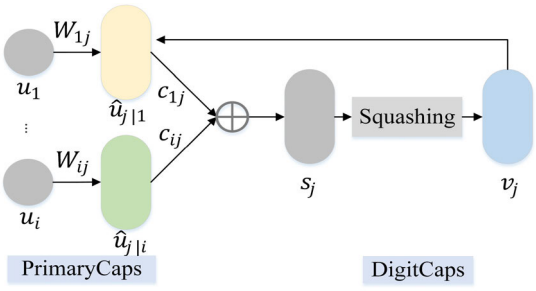

FIGURE 6. Connection process between capsule layers.

through a scale operation, the learned channel weights are multiplied with the original feature maps to obtain the attention feature maps as the output of the SE block. The calculation formulas are shown in (5) - (7):

$$z_C = F_{sq}(u_C) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} u_C(i,j) \tag{5}$$

$$s = F_{ex}(z, W) = \sigma(g(z, W)) = \sigma(W_2 \delta(W_1 z)) \tag{6}$$

$$\widetilde{x_C} = F_{scale}(u_C, s_C) = s_C \cdot u_C \tag{7}$$

3) PRIMARYCAPS LAYER

After the SE block, we take each feature map with its corresponding attention weight as the input of PrimaryCaps. The PrimaryCaps layer is different from the ordinary convolution layer. According to its definition, after this layer, we can obtain capsules, and the capsules can also be called vectors, which can store much information.

4) DIGITCAPS LAYER

The DigitCaps layer is used to store Ponzi and non-Ponzi capsules. The final output is represented by the vector. A squashing function is used by the capsule network. While maintaining the direction of the vector, the length of the output vector is used as the probability of the presence of an entity. The calculation formulas between capsule i and capsule j are shown in (8) - (10):

$$\hat{u}_{j|i} = W_{ij} u_i \tag{8}$$

$$s_j = \sum_i c_{ij} \hat{u}_{j|i} \tag{9}$$

$$v_j = \frac{\|s_j\|^2}{1 + \|s_j\|^2} \frac{s_j}{\|s_j\|} \tag{10}$$

where W_{ij} represents the weight matrix, representing the relationship between capsule i and capsule j, and $\hat{u}_{j|i}$ means the prediction that the i-th low-level capsule constitutes the j-th high-level capsule. c_{ij} is the coupling coefficient obtained through dynamic routing. The output v_j is judged by the result of the final squashing function. The process between capsule layers is shown in Figure 6.

IV. EXPERIMENTAL EVALUATION

A. DATASET

This paper uses the public Ponzi scheme dataset [17], which has manually-checked code logic to determine if the contract is a Ponzi scheme. The creators marked 3590 non-Ponzi contracts and 200 Ponzi contracts. The bytecodes and ABIs of contracts are obtained through the API interface provided by Etherscan [6], and a contract is removed if its bytecode or ABI is null. There are 27 abnormal non-Ponzi contracts, and finally 3563 non-Ponzi contracts are selected as the data for this paper. Then, the features of the downloaded bytecodes and ABIs are extracted and processed, and they are visualized and converted into RGB images to obtain a relatively imbalanced dataset. After the augmentation of image data, Ponzi images are obtained, forming a more balanced dataset. For this experiment, 70% of the data are selected randomly as the training set, 10% of the data are selected as the validation set, and the remainder of the data are used as the test set.

B. METRICS

For single-label image classification problems, accuracy, precision, recall, F1 score, ROC and AUC are usually chosen as evaluation metrics. Comprehensively considering the experimental data, this paper selects the first four metrics for measuring the effectiveness of the proposed method. The specific calculation formulas are as shown in (11) - (14):

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{11}$$

$$Precision = \frac{TP}{TP + FP} \tag{12}$$

$$Recall = \frac{TP}{TP + FN} \tag{13}$$

$$F1 = \frac{2 * precision * recall}{precision + recall} \tag{14}$$

Among them, TP stands for a Ponzi contract that is actually judged as a Ponzi contract; FN denotes a real Ponzi contract that is judged as a non-Ponzi contract; FP stands for a non-Ponzi contract that is misjudged as a Ponzi contract; TN means that a non-Ponzi contract is correctly judged as a non-Ponzi contract.

C. EXPERIMENTAL SETUP

This paper incorporates the SE block and capsule network to complete the task of detecting Ponzi contracts on Ethereum. The image widths of the model inputs may affect the results of the experiments. According to the image width recommendations for the various file sizes proposed in the paper in [20], the image width for files less than 10 KB is generally selected as 32. Therefore, we uniformly use $32 * 32$ RGB images as the input of the model.

In terms of experimental settings, we use the Python language to build our method. For feature visualization, we employ NumPy, Pandas, OpenCV and other Python packages for image feature extraction and processing. The models are built by using Keras and TensorFlow. During the experiment, the parameters of the model affect the training results. Considering the actual situation regarding the type of detection task, the number of samples, the memory size of

TABLE 1. The parameters in the proposed method.

Parameter	Value
Batch Size	64
Epoch	15
Learning Rate	0.001
Weight Decay	0.0005
Routing Iterations	3

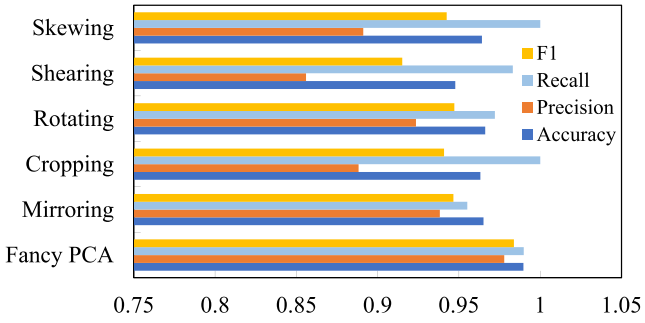

FIGURE 7. Comparison with different data augmentation methods.

the CPU and the time consumption of the training process, we select appropriate parameters, as shown in Table 1.

D. EXPERIMENTAL RESULTS AND ANALYSIS

1) COMPARISON WITH DIFFERENT DATA AUGMENTATION METHODS

Imbalanced data is the primary problem that needs to be resolved. After data screening, the ratio of Ponzi contracts to non-Ponzi contracts is approximately 1:18, and there is a serious data imbalance. Existing data augmentation methods include perspective skewing, elastic distortions, rotating, shearing, cropping, mirroring, etc. The fancy PCA method mainly realizes image augmentation by changing the intensity of the RGB channel in the training image. The following figure shows the experimental results of image augmentation using skewing, shearing, rotating, cropping, mirroring, and fancy PCA methods.

As shown in Figure 7, the recall of fancy PCA is lower those that of cropping and skewing, but the best results are obtained for other metrics. The F1 score is increased by 4.30% and 4.14% compared to those of cropping and skewing, respectively. The accuracy is approximately 2% higher than those of the other five methods. After analyzing the experimental results and performing comprehensive measurements, we select the fancy PCA method for data augmentation, as it can achieve the best effect and has a positive influence on the classification results.

2) PERFORMANCE EVALUATION OF DATASETS WITH DIFFERENT RATIOS

When the number of Ponzi contracts is insufficient, this seriously affects the classification effect of the model, resulting in large classification errors and an extremely low detection rate with respect to Ponzi contracts. By changing the value of the standard deviation α_i in fancy PCA, we can obtain many new Ponzi contract images. In real life, the number of non-Ponzi contracts is generally greater than that of Ponzi contracts. Under the assumption that this condition is met, we explore the impact of different ratios of non-Ponzi contracts to Ponzi contracts on the experimental results. The experimental results are shown in Table 2.

TABLE 2. Performance evaluation of datasets with different ratios.

Ponzi: Non-ponzi	Precision	Recall	F1 score
1:18 (Original Dataset)	0.5000	0.4736	0.4865
1:5	0.6785	0.8535	0.7180
1:2	0.9779	0.9898	0.9838

TABLE 3. Experimental results obtained by different models.

Model	Accuracy	Precision	Recall	F1 score
Random Forest	0.9346	0.9643	0.7671	0.8544
XGBoost	0.9776	0.9957	0.9200	0.9563
AdaBoost	0.8945	0.9809	0.6160	0.7568
LightGBM	0.9289	0.9505	0.8617	0.8956
VGGNet	0.9826	0.9782	0.9662	0.9721
ResNet	0.9642	0.9747	0.9496	0.9605
MiniGoogLeNet	0.9768	0.9829	0.9664	0.9739
MobileNet	0.9502	0.9645	0.9113	0.9337
DenseNet	0.9361	0.9541	0.9132	0.9284
SE-CapsNet	0.9897	0.9779	0.9898	0.9838

For imbalanced data, the accuracy metric is not applicable, because the model may be biased towards the majority class, it can easily achieve a high accuracy. In this part, we use weighted average to calculate the precision, recall, and F1 score of extremely imbalanced data. From the table, we can see that without image augmentation, the original dataset is used for Ponzi contract detection, and the model almost predicts most of the test set data as non-Ponzi contracts, resulting in very low precision, recall and F1 score. When the ratio between the two contract types is approximately 1:5. At this time, the F1 score rise steadily, reaching 71.80%. In the end, when the ratio of Ponzi contracts to non-Ponzi contracts is 1:2, all evaluation metrics achieve relatively good results. At the same time, it's consistent with the fact that there are more non-Ponzi contracts than Ponzi contracts. From this, we can see that the expansion of the dataset to achieve balanced data and the selection of an appropriate data ratio are critical to the effectiveness of the experimental results.

3) COMPARISON WITH EXPERIMENTAL RESULTS OBTAINED BY DIFFERENT MODELS

Next, we consider how to verify whether different models have an impact on the detection of Ponzi schemes. Nine models, Random Forest, XGBoost, AdaBoost, LightGBM, VGGNet, ResNet, MiniGoogLeNet, MobileNet, and DenseNet, are selected. We can see the detection results obtained by different models in the comparative experiment in Table 3.

TABLE 4. The SE-CapsNet model compared with the CapsNet.

Model	Accuracy	Precision	Recall	F1 score
CapsNet	0.9843	0.9898	0.9686	0.9787
SE-CapsNet	0.9897	0.9779	0.9898	0.9838

As seen from the above table, the image-based scam detection method has achieved good results in both machine learning and deep learning methods. The XGBoost performs better in machine learning models. The detection method based on deep learning can obtain a higher F1 score, and this means that deep learning can be applied to Ponzi contract detection.

However, due to the complex structures of models such as DenseNet and MobileNet, under the same training conditions, the results are not yet optimal. In particular, SE-CapsNet yields good experimental results in most evaluation metrics. The accuracy is 0.71% higher than that of VGGNet, while the F1 score is also improved by 0.99% compared to that of MiniGoogLeNet. The performance improvement is probably due to the architecture of the SE-CapsNet. The SE-CapsNet model can not only retain a large amount of information such as position, but can also highlight the key point of channel information through the SE block. To verify whether the introduction of the SE block has an impact on the effectiveness of the model, in the next step, we would like to compare the SE-CapsNet model with only CapsNet to verify its detection effect. Table 4 shows the results of this experiment.

An analysis of the experimental results shows that the accuracy of the SE-CapsNet model in the experiment is 98.97%, and the F1 score reaches 98.38%. SE-CapsNet obtains desirable classification results. Compared with the CapsNet model alone, the accuracy is improved by 0.54%, while the F1 score is increased by 0.51%. Therefore, we can conclude that the introduction of the SE block has a certain role in promoting the classification performance of the proposed method for detecting Ponzi schemes contracts.

4) COMPARISON BETWEEN DIFFERENT DETECTION METHODS

Based on the research of the paper in [17], this paper proposes a variety of features based on contract information. In recent years, the amount of related research has gradually increased. However, some researchers use both contract and transaction features for detecting. The use of transaction features cannot achieve the purpose of discovering the Ponzi contracts in time. Below, we only use contract information to compare our proposed method with the corresponding methods in prior work.

As shown in Table 5, our method can be used to detect Ponzi contracts as soon as they are deployed to the blockchain. It has an F1 score of 98%, which is improved from 82%, 95% and 96% in prior works. Through experiments, we can see that our proposed method is not only effective for early Ponzi scheme detection but also improves the accuracy of detection.

TABLE 5. Performance comparison only based on contract information.

Method	Precision	Recall	F1 score
[17]	0.94	0.73	0.82
[18]	0.98	0.94	0.96
[39]	0.98	0.93	0.95
Ours	0.98	0.99	0.98

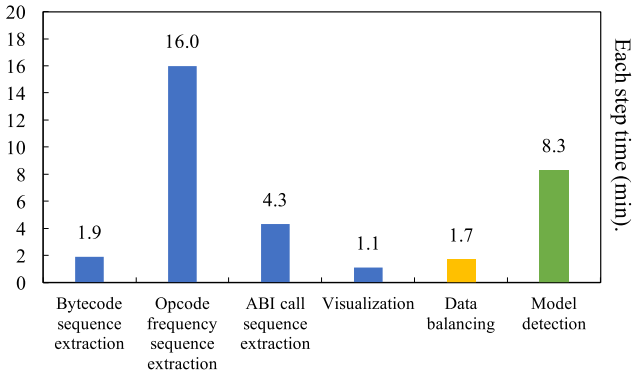

FIGURE 8. The time consumption of each step.

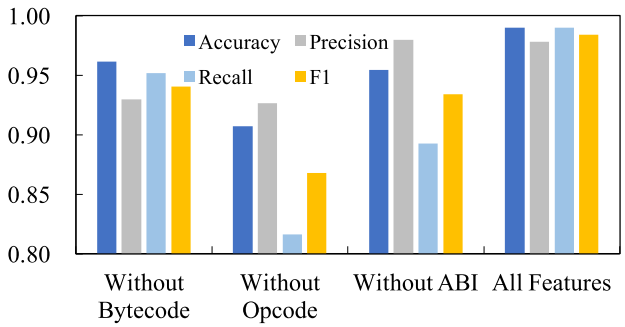

FIGURE 9. Performance comparison with different features.

5) TIME CONSUMPTION

To further explore the efficiency of the experiment, we recorded the time consumption of each step: feature visualization, data balancing and model detection. The experimental results are shown in Figure 8.

Through the above figure, we can see that blue, yellow and green bars represent the time consumption of feature visualization, data balancing and model detection, respectively. The processing time of the data balancing module is the shortest, while the feature visualization component takes up a large amount of time. Among them, the time needed to extract the opcodes and convert them into the opcode frequency sequences is about 16 minutes. We know from further calculations that the time consumption of processing each contract is approximately 0.39 s, among which the feature visualization portion takes 0.27 s.

6) INFLUENCES OF DIFFERENT FEATURES ON THE EXPERIMENTAL RESULTS

To clarify the effects of the bytecode sequence, the opcode frequency sequence and the ABI call sequence on the performance of the experiment, we set the pixels of the above

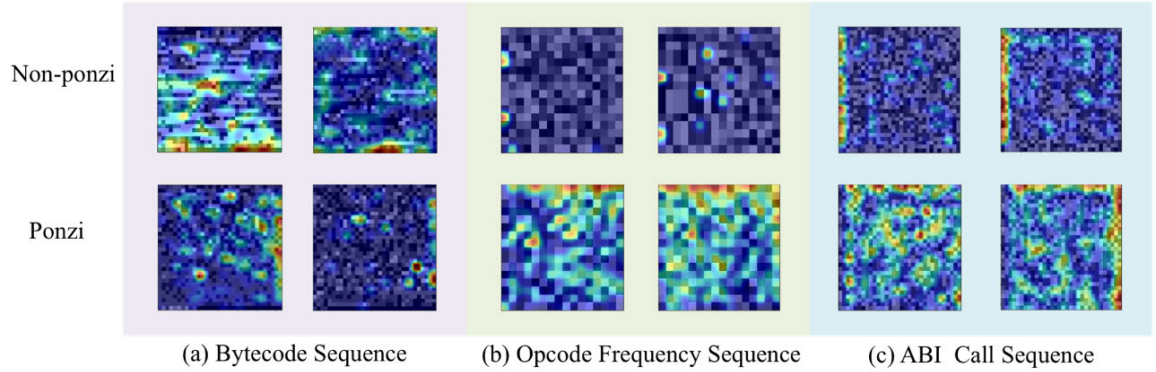

FIGURE 10. The Ponzi and non-Ponzi images obtained by Grad-CAM calculation.

three types of feature images to 0 respectively to obtain all-black images. And then combine the black images with the remaining feature images to form the input of the SE-CapsNet model. Next, we determine which feature is most important to the experimental results. The following is a comparison graph of the experiments using different features.

As shown in Figure 9, Ponzi contracts detection without the opcode frequency sequence feature achieves poor experimental results. The F1 score is 86.78%, and the accuracy reaches 90.71%. It can be shown that the opcode frequency sequence can effectively enhance the experimental performance of the model. Subsequently, we can see that there is not much difference between the results of the experiments without the bytecode sequence and without the ABI call sequence. Their accuracy rates still exceed 95%, and although these two features are not the most important features affecting the performance of the model, they are still an indispensable part of the method proposed in this paper.

7) FEATURE INTERPRETABILITY ANALYSIS

As the research on interpretability deepened, numerous interpretable models have emerged, making the mysterious black box of neural networks easy for humans to understand to some extent. Grad-CAM [40], which is a technology that can provide the visual interpretation, is mainly adopted in this paper. Using this method, the pixels that influence the category can be obtained and highlighted on the original image.

To analyze the extracted features clearly, this paper uses a neural network (CNN) to train the images of the bytecode sequence, the opcode frequency sequence and the ABI call sequence then carries out Grad-CAM calculations, visually displaying the differences between Ponzi contracts and non-Ponzi contracts. The above figures show the graphs of the Ponzi contract and the non-Ponzi contract obtained by Grad-CAM calculations.

From Figure 10, we know that although the three features selected are different, the images generated by the Ponzi contract and the non-Ponzi contract have regularity. From the perspective of the bytecode sequence features, the pixels that affect the classification results of the Ponzi images are mainly concentrated in the right and middle areas, showing an overall sporadic distribution trend. Grad-CAM calculates that the highlighted pixels of non-Ponzi images are mostly concentrated in the tail line. We can see that the main pixels determined using the opcode frequency sequence and the ABI call sequence features are similar. The calculation result of the non-Ponzi images is highlighted in the left area, with regular intervals. Highlighted pixels of Ponzi images can be clearly observed in the first row and the right area. One can see that there are obvious logical differences between Ponzi contracts and non-Ponzi contracts. In a case where a given contract has a source code, it is possible to analyze the end of the source code, the most frequent opcode sequence, and the front part of the ABI call sequence, thereby enabling highly efficient smart contract scam detection.

TABLE 6. Analysis on the detection results of honeypot contract.

Model	Accuracy	Precision	Recall	F1 score
CapsNet	0.9629	0.8805	0.9614	0.9192
SE-CapsNet	0.9767	0.9078	0.9842	0.9444

8) CASE STUDY

To demonstrate whether the proposed method can be used to detect other fraudulent accounts, next, we detect a new type of smart contract called "honeypot" in Ethereum. Scammers deliberately design contracts with a flaw to entice some greedy users to exploit that flaw, thereby draining the funds of users and leading to irreparable losses [41]. This paper examines the 1124 honeypot contract accounts from the HONEYBADGER project [15] as the fraud dataset for the case study. Combined with 3563 benign accounts, after feature visualization, the SE-CapsNet model is used to detect honeypot contracts. The results of the case study are shown in Table 6.

From the experimental results, we can see that the accuracy of the SE-CapsNet model reaches 97.67%, and the F1 score reaches 94.44%. It can be seen from the case study that the

method proposed in this paper can detect not only Ponzi contracts with imbalanced samples but also relatively balanced honeypot contracts, and this has certain research value for the detection of other scam accounts in Ethereum.

V. CONCLUSION

The trend towards using smart contracts to execute scams is becoming increasingly severe. A Ponzi scheme is a typical scam method in Ethereum. It is difficult to detect Ponzi schemes in real time with traditional transaction-based methods. Therefore, this paper only uses contract information for Ponzi schemes detection. The proposed method uses the bytecode and ABI of the contract for detection and analysis to improve upon the limitation resulting from only using the source code of the contract. After feature visualization, SE-CapsNet is used to detect Ponzi schemes in Ethereum. The detection results are enhanced over those of other detection methods. However, there are two shortcomings to the method in this paper. One is that the training time required for the model is relatively long, and the other is that the number of available Ponzi contracts is insufficient. In the future, we may consider continuing to improve the experiment, collecting additionally Ponzi samples, appropriately increasing the mapping relationships between features to enrich the feature space, and optimizing the detection process in this paper (for example, using a two-sample test for detection [42]). At the same time, we can extend the applicability of the proposed method to other types of fraud detection, such as ransomware and fake token sales.

ACKNOWLEDGMENT

The authors would like to thank reviewers for their precious remarks and comments.

REFERENCES

[1] J. Wu, J. Liu, W. Chen, H. Huang, Z. Zheng, and Y. Zhang, "Detecting mixing services via mining bitcoin transaction network with hybrid motifs," 2020, *arXiv:2001.05233*. [Online]. Available: http://arxiv.org/abs/2001.05233

[2] Chainalysis. (Jan. 2020). *The 2020 State of Crypto Crime*. [Online]. Available: https://blog.chainalysis.com/reports/cryptocurrency-crime-2020-report

[3] T. Moore, J. Han, and R. Clayton, "The postmodern Ponzi scheme: Empirical analysis of high-yield investment programs," in *Proc. 16th Int. Conf. Financial Cryptogr. Data Secur.*, Mar. 2012, pp. 41–56.

[4] Q. Bai, C. Zhang, Y. Xu, X. Chen, and X. Wang, "Evolution of ethereum: A temporal graph perspective," 2020, *arXiv:2001.05251*. [Online]. Available: http://arxiv.org/abs/2001.05251

[5] S. Rouhani and R. Deters, "Security, performance, and applications of smart contracts: A systematic survey," *IEEE Access*, vol. 7, pp. 50759–50779, Apr. 2019.

[6] (Jun. 2019). *Etherscan*. [Online]. Available: https://etherscan.io/

[7] W. Joon-Wie Tann, X. Jie Han, S. Sen Gupta, and Y.-S. Ong, "Towards safer smart contracts: A sequence learning approach to detecting security threats," 2018, *arXiv:1811.06632*. [Online]. Available: http://arxiv.org/abs/1811.06632

[8] G. Tian, Q. Wang, Y. Zhao, L. Guo, Z. Sun, and L. Lv, "Smart contract classification with a bi-LSTM based approach," *IEEE Access*, vol. 8, pp. 43806–43816, Mar. 2020.

[9] S. Farrugia, J. Ellul, and G. Azzopardi, "Detection of illicit accounts over the ethereum blockchain," *Expert Syst. Appl.*, vol. 150, Jul. 2020, Art. no. 113318.

[10] L. Bian, L. Zhang, K. Zhao, and F. Shi, "Ethereum malicious account detection method based on LightGBM," *Netinfo Secur.*, vol. 20, no. 4, pp. 73–80, Apr. 2020.

[11] J. Wu, Q. Yuan, D. Lin, W. You, W. Chen, C. Chen, and Z. Zheng, "Who are the phishers? Phishing scam detection on ethereum via network embedding," 2019, *arXiv:1911.09259*. [Online]. Available: http://arxiv.org/abs/1911.09259

[12] K. Toyoda, T. Ohtsuki, and P. T. Mathiopoulos, "Identification of high yielding investment programs in bitcoin via transactions pattern analysis," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2017, pp. 1–6.

[13] M. Bartoletti, B. Pes, and S. Serusi, "Data mining for detecting bitcoin ponzi schemes," in *Proc. Crypto Valley Conf. Blockchain Technol. (CVCBT)*, Jun. 2018, pp. 75–84.

[14] M. Bartoletti, S. Carta, T. Cimoli, and R. Saia, "Dissecting ponzi schemes on ethereum: Identification, analysis, and impact," *Future Gener. Comput. Syst.*, vol. 102, pp. 259–277, Jan. 2020.

[15] C. Ferreira Torres, M. Steichen, and R. State, "The art of the scam: Demystifying honeypots in ethereum smart contracts," 2019, *arXiv:1902.06976*. [Online]. Available: http://arxiv.org/abs/1902.06976

[16] T. Chen, T. Hu, J. Chen, X. Zhang, Z. Li, Y. Zhang, X. Luo, A. Chen, K. Yang, B. Hu, T. Zhu, and S. Deng, "DataEther: Data exploration framework for ethereum," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2019, pp. 1369–1380.

[17] W. Chen, Z. Zheng, E. C.-H. Ngai, P. Zheng, and Y. Zhou, "Exploiting blockchain data to detect smart ponzi schemes on ethereum," *IEEE Access*, vol. 7, pp. 37575–37586, Mar. 2019.

[18] E. Jung, M. Le Tilly, A. Gehani, and Y. Ge, "Data mining-based ethereum fraud detection," in *Proc. IEEE Int. Conf. Blockchain (Blockchain)*, Jul. 2019, pp. 266–273.

[19] A. Kartel, E. Novikova, and A. Volosiuk, "Analysis of visualization techniques for malware detection," in *Proc. IEEE Conf. Russian Young Researchers Electr. Electron. Eng. (EIConRus)*, Jan. 2020, pp. 337–340.

[20] L. Nataraj, S. Karthikeyan, G. Jacob, and B. S. Manjunath, "Malware images: Visualization and automatic classification," in *Proc. IEEE 8th Int. Symp. Vis. Cyber Secur.*, Jul. 2011, pp. 1–7.

[21] Z. Cui, F. Xue, X. Cai, Y. Cao, G.-G. Wang, and J. Chen, "Detection of malicious code variants based on deep learning," *IEEE Trans. Ind. Informat.*, vol. 14, no. 7, pp. 3187–3196, Jul. 2018.

[22] Z. Cui, L. Du, P. Wang, X. Cai, and W. Zhang, "Malicious code detection based on CNNs and multi-objective algorithm," *J. Parallel Distrib. Comput.*, vol. 129, pp. 50–58, Jul. 2019.

[23] H. Naeem, B. Guo, M. R. Naeem, F. Ullah, H. Aldabbas, and M. S. Javed, "Identification of malicious code variants based on image visualization," *Comput. Electr. Eng.*, vol. 76, pp. 225–237, Jun. 2019.

[24] B. Sun, P. Zhang, M. Cheng, X. Li, and Q. Li, "Malware detection method based on enhanced code images," *J. Tsinghua Univ. (Sci. Technol.)*, vol. 60, no. 5, pp. 386–392, Apr. 2020.

[25] Y. Fang, Y. Gao, F. Jing, and L. Zhang, "Android malware familial classification based on DEX file section features," *IEEE Access*, vol. 8, pp. 10614–10627, Jan. 2020.

[26] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-excitation networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 8, pp. 2011–2023, Aug. 2020.

[27] H. Yu, "Research on classification of retinal diseases based on SE-block," M.S. thesis, College Softw., Jilin Univ., Jilin, China, 2019.

[28] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Proc. 31st Neural Inf. Process. Syst.*, Dec. 2017, pp. 3856–3866.

[29] Z. Dong and S. Lin, "Research on image classification based on capsnet," in *Proc. IEEE 4th Adv. Inf. Technol., Electron. Autom. Control Conf. (IAEAC)*, Dec. 2019, pp. 1023–1026.

[30] R. Mukhometzianov and J. Carrillo, "CapsNet comparative performance evaluation for image classification," 2018, *arXiv:1805.11195*. [Online]. Available: http://arxiv.org/abs/1805.11195

[31] S. Bonheur, D. Štern, C. Payer, M. Pienn, H. Olschewski, and M. Urschler, "Matwo-CapsNet: A multi-label semantic segmentation capsules network," in *Proc. 22nd Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.*, Oct. 2019, pp. 664–672.

[32] C. Xiang, L. Zhang, Y. Tang, W. Zou, and C. Xu, "MS-CapsNet: A novel multi-scale capsule network," *IEEE Signal Process. Lett.*, vol. 25, no. 12, pp. 1850–1854, Dec. 2018.

[33] X. Cheng, J. He, J. He, and H. Xu, "Cv-CapsNet: Complex-valued capsule network," *IEEE Access*, vol. 7, pp. 85492–85499, Jun. 2019.

[34] (Apr. 2020). *Solidity*. [Online]. Available: https://solidity.readthedocs.io/en/v0.6.6/
[35] (Jul. 2020). *Ethereum-Dasm*. [Online]. Available: https://github.com/tintinweb/ethereum-dasm
[36] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *Proc. 34th Annu. ACM Symp. Theory Comput. (STOC)*, 2002, pp. 380–388.
[37] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proc. 31st Int. Mach. Learn.*, Jun. 2014, pp. 1188–1196.
[38] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. 25th Neural Inf. Process. Syst.*, Dec. 2012, pp. 1097–1105.
[39] J. Peng and G. Xiao, "Detection of smart Ponzi schemes using opcode," in *Proc. 2nd BlockSys*, Aug. 2020, pp. 192–204.
[40] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 336–359, Feb. 2020.
[41] R. Camino, C. F. Torres, M. Baden, and R. State, "A data science approach for detecting honeypots in ethereum," in *Proc. IEEE Int. Conf. Blockchain Cryptocurrency (ICBC)*, May 2020, pp. 1–9.
[42] R. Gao, F. Liu, J. Zhang, B. Han, T. Liu, G. Niu, and M. Sugiyama, "Maximum mean discrepancy is aware of adversarial attacks," 2020, *arXiv:2010.11415*. [Online]. Available: http://arxiv.org/abs/2010.11415

LINGYU BIAN received the B.S. degree in network engineering from Xinjiang University, in 2018, where she is currently pursuing the master's degree with the School of Cyber Science and Engineering, College of Information Science and Engineering. Her research interests include blockchain security and data analysis.

LINLIN ZHANG received the Ph.D. degree in computer software and theory from Wuhan University, in 2009. She is currently an Associate Professor with the School of Cyber Science and Engineering, College of Information Science and Engineering, Xinjiang University. Her current research interests include software security, big data analysis, and so on.

KAI ZHAO received the Ph.D. degree in computer software and theory from Wuhan University, in 2011. He is currently an Associate Professor with the School of Cyber Science and Engineering, College of Information Science and Engineering, Xinjiang University. His current research interests include software security, big data analysis, medical information processing, and so on.

HAO WANG received the B.S. degree in internet of things engineering from the China University of Petroleum, East China, in 2018. He is currently pursuing the master's degree with the School of Software, Xinjiang University. His research interest includes blockchain applications.

SHENGJIA GONG received the B.S. degree in computer science and technology from Huanggang Normal University, in 2018. He is currently pursuing the master's degree with the School of Cyber Science and Engineering, College of Information Science and Engineering, Xinjiang University. His research interest includes blockchain applications.